

# **SC-703 Desarrollo de Aplicaciones Móviles**

**ICO 2022  
SEMANA 2-3  
Room/ SQLite /  
Mejorando Diseños**

# • Agenda



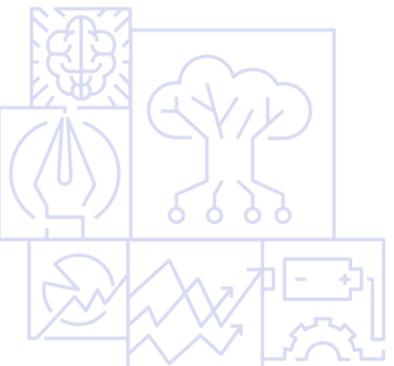
Configuración del  
proyecto



ROOM / SQLite



Mejorando las vistas



- AJUSTAR EL PROYECTO DE LA SEMANA PASADA  
(SUBIDO POR EL PROFESOR)



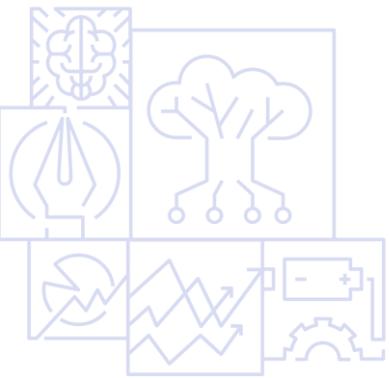
Si desea tomar el proyecto desarrollado por el profesor y continuarlo esta semana



En semana 02 está el proyecto subido. Se debe descargar y utilizar el “Google-services.json” de su cuenta de Firebase



Luego de esto ya está listo para ver la clase de hoy



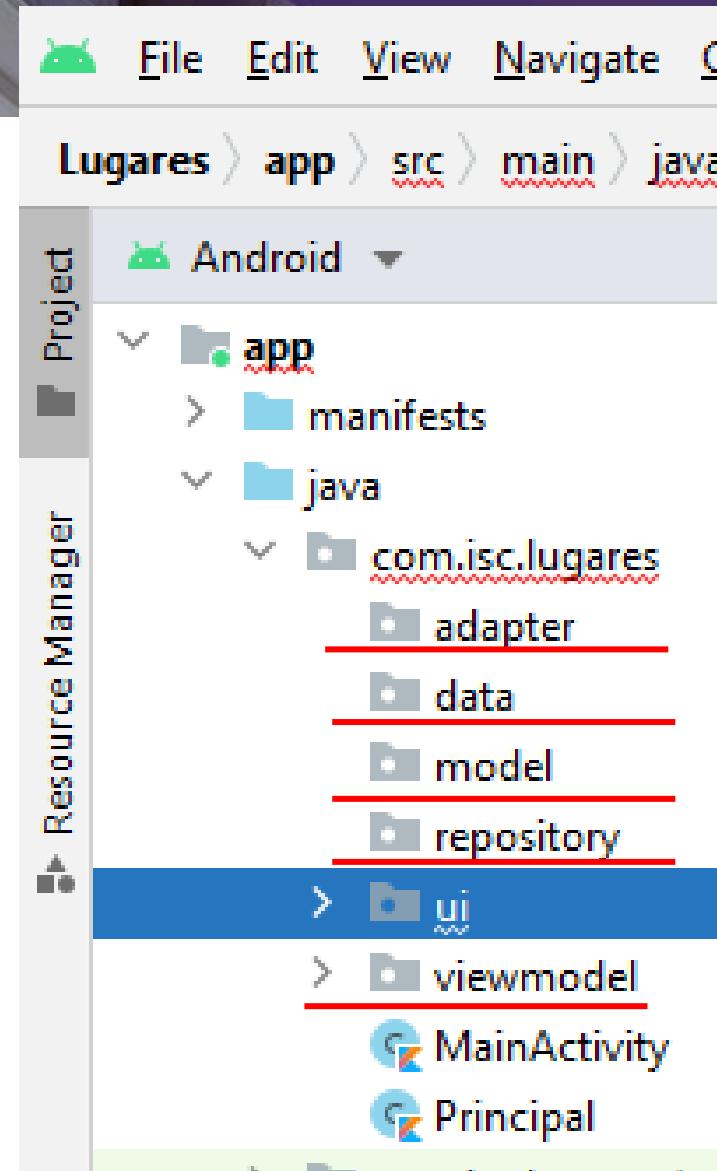
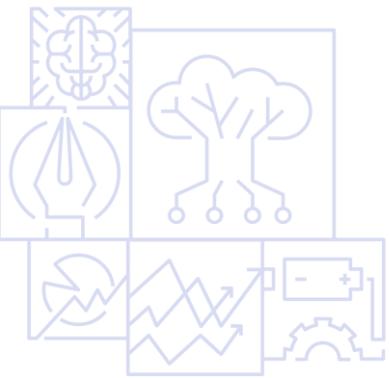


A photograph of a modern university building at dusk. The building, labeled 'Edificio A', features a curved facade with large windows and a prominent blue and white spiral staircase. A sign in front of the entrance reads: 'EDIFICIO A', 'Rectoría', '• Aulas 31-56', and '• Oficinas Administrativas'. The sky is a gradient of purple and orange, and a bus stop is visible on the right side of the street.

## • MEJORANDO LA ARQUITECTURA DEL PROYECTO

- SE CREAN LOS SIGUIENTES PAQUETES EN EL PROYECTO LUGARES

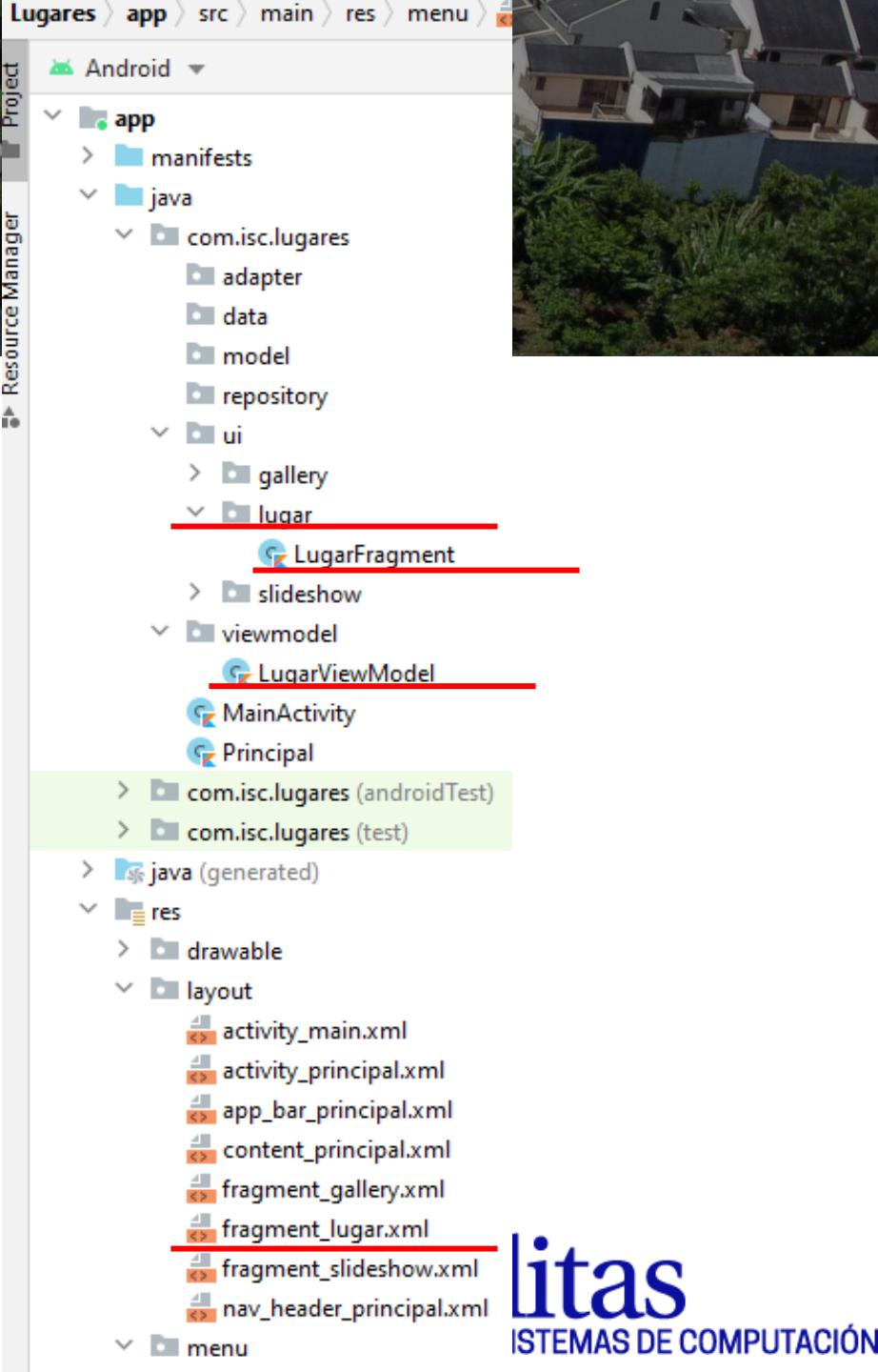
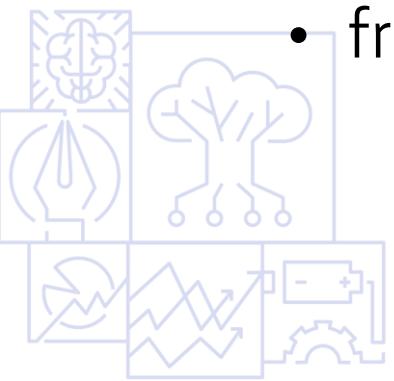
adapter  
data  
model  
repository  
viewmodel



# Otros ajustes



1. Renombramos el paquete:
  - home → lugar
2. Renombramos las clases:
  - HomeFragment → LugarFragment
  - HomeViewModel → LugarViewModel
3. Movemos:
  - LugarViewModel al paquete viewmodel
4. Renombramos el archivo:
  - fragment\_home.xml → fragment\_lugar.xml



- Se corrigen errores y se actualizan nombres de variables

LugarFragment.kt

```

10 import androidx.lifecycle.ViewModelProvider
11 import com.isc.lugares.databinding.FragmentHomeBinding
12 import com.isc.lugares.viewmodel.LugarViewModel
13
14 class LugarFragment : Fragment() {
15
16     private lateinit var homeViewModel: LugarViewModel
17     private var _binding: FragmentHomeBinding? = null
18
19     // This property is only valid between onCreateView and
20     // onDestroyView.
21     private val binding get() = _binding!!
22
23     override fun onCreateView(
24         inflater: LayoutInflater,
25         container: ViewGroup?,
26         savedInstanceState: Bundle?
27     ): View? {
28         homeViewModel =
29             ViewModelProvider(owner: this).get(LugarViewModel::class.java)
30
31         _binding = FragmentHomeBinding.inflate(inflater, container, false)
32         val root: View = binding.root

```



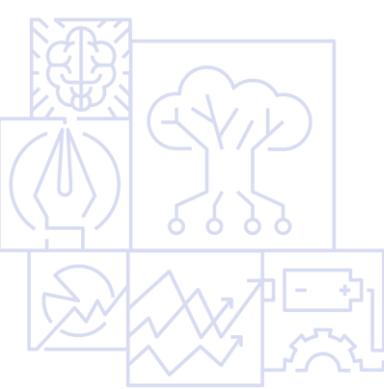
LugarFragment.kt

```

11 import com.isc.lugares.databinding.FragmentLugarBinding
12 import com.isc.lugares.viewmodel.LugarViewModel
13
14 class LugarFragment : Fragment() {
15
16     private lateinit var lugarViewModel: LugarViewModel
17     private var _binding: FragmentLugarBinding? = null
18     private val binding get() = _binding!!
19
20     override fun onCreateView(
21         inflater: LayoutInflater,
22         container: ViewGroup?,
23         savedInstanceState: Bundle?
24     ): View? {
25         lugarViewModel =
26             ViewModelProvider(owner: this).get(LugarViewModel::class.java)
27
28         _binding = FragmentLugarBinding.inflate(inflater, container, attach
29         val root: View = binding.root
30
31         val textView: TextView = binding.textHome
32         lugarViewModel.text.observe(viewLifecycleOwner, Observer { it: String!
33             textView.text = it
34 })

```

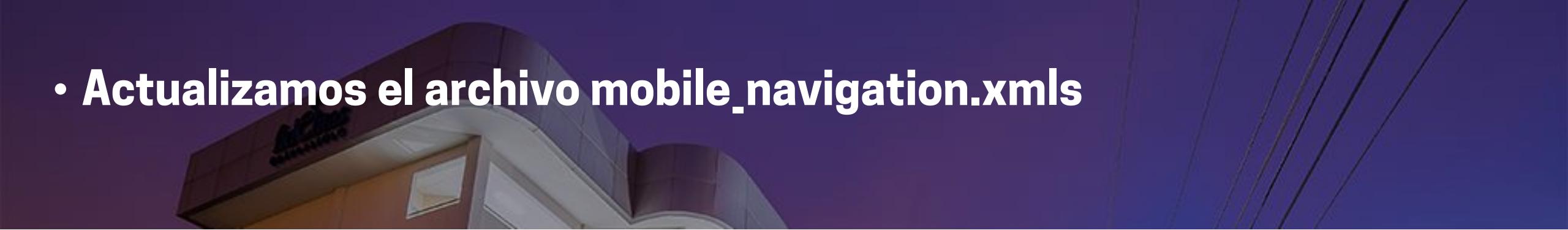
# • Actualizamos archivo string.xml



```
strings.xml x
Edit translations for all locales in the translations editor.

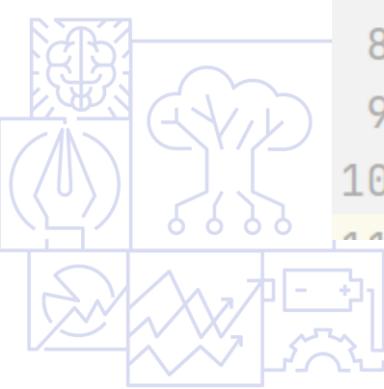
1 <resources>
2     <string name="app_name">Lugares</string>
3     <string name="msg_datos">Requiere texto en los ca...
4     <string name="msg_fallo">Falló la autenticación..
5     <string name="title_activity_principal">Principal...
6     <string name="navigation_drawer_open">Open navigat...
7     <string name="navigation_drawer_close">Close navi...
8     <string name="nav_header_title">Android Studio</s...
9     <string name="nav_header_subtitle">android.studio(...
10    <string name="nav_header_desc">Navigation header<...
11    <string name="action_settings">Settings</string>
12    <string name="menu_lugar">Lugares</string>
13    <string name="menu_gallery">Gallery</string>
14    <string name="menu_slideshow">Slideshow</string>
15 </resources>
```

- Actualizamos el archivo `mobile_navigation.xml`



```
mobile_navigation.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <navigation xmlns:android="http://schemas.android.com/apk/res/android"
3   xmlns:app="http://schemas.android.com/apk/res-auto"
4   xmlns:tools="http://schemas.android.com/tools"
5   android:id="@+id/mobile_navigation"
6   app:startDestination="@+id/nav_lugar">
7
8   <fragment
9     android:id="@+id/nav_lugar"
10    android:name="com.isc.lugares.ui.lugar.LugarFragment"
11    android:label="@string/menu_lugar"
12    tools:layout="@layout/fragment_lugar" />
13
```

- Se modifica el menú activity\_main\_Drawer.xml



```
activity_main_drawer.xml <?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/
    xmlns:tools="http://schemas.android.com/tools"
    tools:showIn="navigation_view">

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_lugar"
            android:icon="@drawable/ic_menu_camera"
            android:title="@string/menu_lugar" />
    
```

# • Se modifica la clase Principal.kt



```
Principal.kt
32     val navController = findNavController(R.id.nav_controller)
33     // Passing each menu ID as a set of IDs because
34     // menu should be considered as top level destination
35     appBarConfiguration = AppBarConfiguration(
36         setOf(
37             R.id.nav_lugar, R.id.nav_gallery, R.id.nav_slideshow
38         ), drawerLayout
39     )
40     setupActionBarWithNavController(navController)
41     navView.setupWithNavController(navController)
42 }
43 }
```



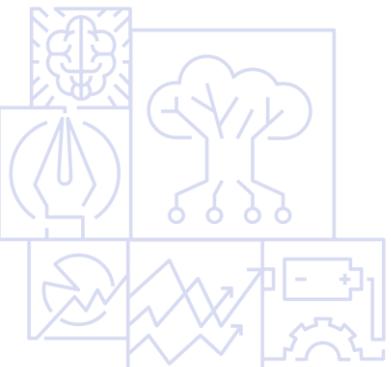
- Ya se tiene el proyecto listo en arquitectura, ahora se prepara para nuevo material

# Uso de Room sobre SQLite Base de datos local en el celular

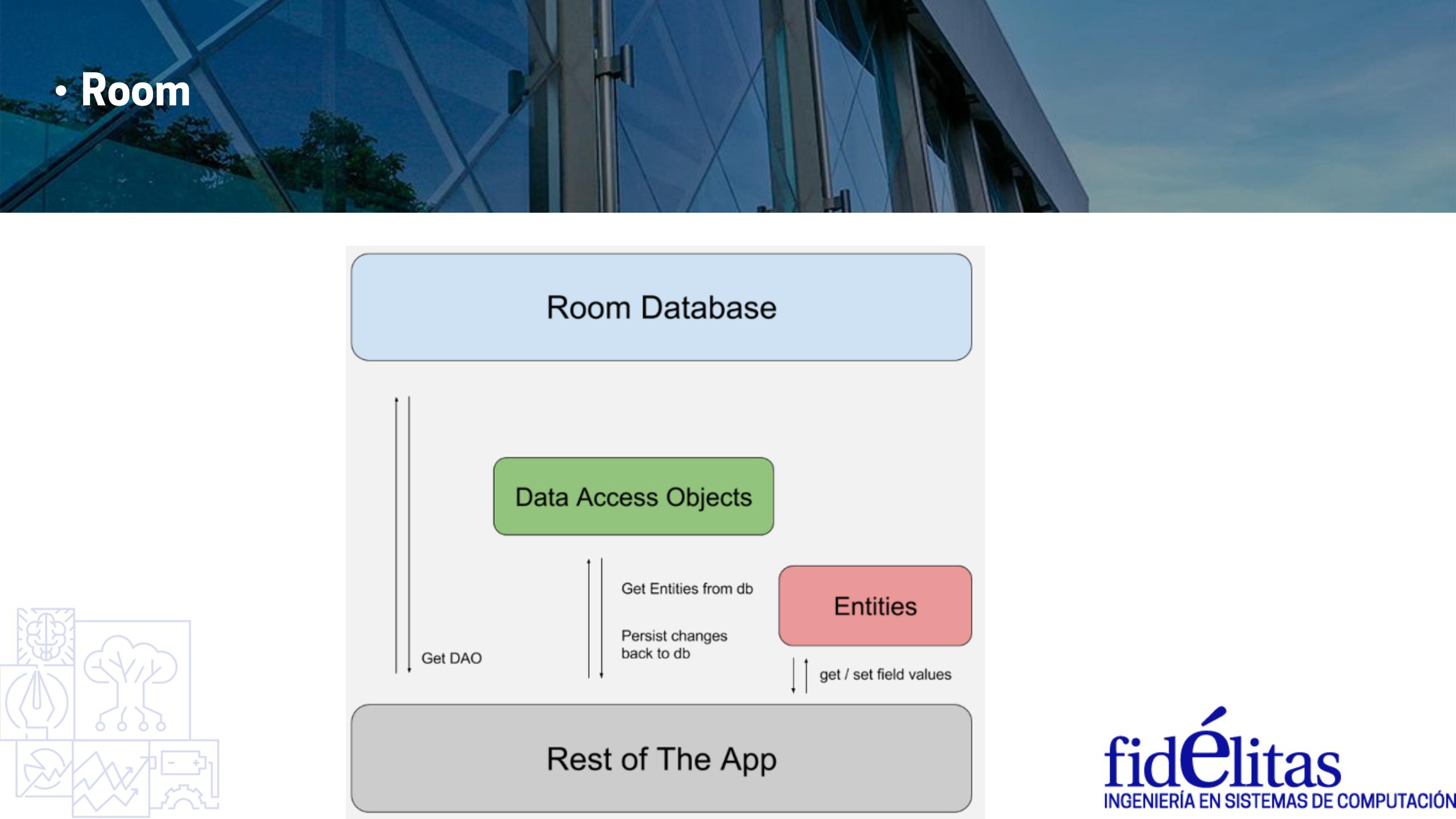
Verificación de las sentencias SQL en tiempo de compilación

Menor repetición de código (boilerplate code)

Fácilmente integrable con otras componentes de la arquitectura



- Room



- Anotaciones



---

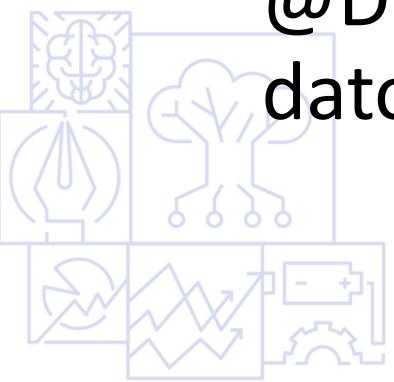
@Entity: representa una tabla dentro de la base de datos, es definida dentro de una clase, no tienen lógica, sólo atributos

---

@Dao: Es la parte responsable de definir las funciones para accesar la base de datos, donde se colocan las sentencias.

---

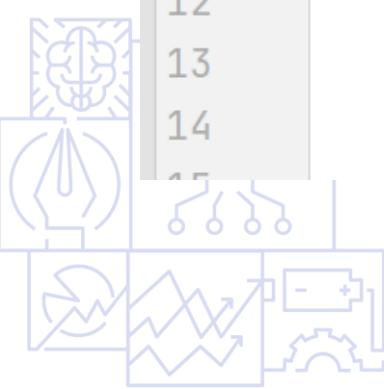
@Database: contiene la definición general de la base de datos, lo de más bajo nivel.



- Actualización de archivo build.gradle



# • Actualizamos el build.gradle (project)



```
build.gradle (Lugares) ×
You can use the Project Structure dialog to view and edit your project configuration

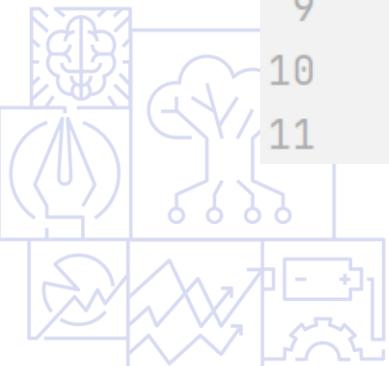
4     google()
5     mavenCentral()
6 }
7 dependencies {
8     classpath "com.android.tools.build:gradle:7.0.2"
9     classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:1.5.31"
10    classpath 'com.google.gms:google-services:4.3.10'
11
12    //Para poder pasar argumentos en la navegación entre fragmentos
13    classpath "androidx.navigation:navigation-safe-args-gradle-plugin:2.3.5"
14
15
```

## • build.gradle (module)



```
build.gradle (:app) X
You can use the Project Structure dialog to view and edit your project configuration

1 plugins {
2     id 'com.android.application'
3     id 'kotlin-android'
4     id 'com.google.gms.google-services'
5     id 'kotlin-parcelize'
6 }
7 apply plugin: 'kotlin-kapt'
8 apply plugin: "androidx.navigation.safeargs" //Para pasar los parámetros...
9
10 //Para generar un enlace directo entre la vista (xml) y el código...
11 android.buildFeatures.viewBinding true
```



# • Continuar archiv

```
build.gradle (:app) X
You can use the Project Structure dialog to view and edit your project configuration
58     androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
59
60     //Se agregan por @Entity de Lugar...
61     annotationProcessor 'androidx.room:room-compiler:2.3.0'
62     kapt "androidx.room:room-compiler:2.3.0"
63     implementation "androidx.room:room-ktx:2.3.0"
64
65     //Navegacion entre fragmentos
66     implementation "androidx.navigation:navigation-fragment-ktx:2.3.5"
67     implementation "androidx.navigation:navigation-ui-ktx:2.3.5"
68     implementation "androidx.navigation:navigation-dynamic-features-fragment:2.3.5"
69
70     //Para clases más adelante...
71     // Esto es para la parte de gps y camara
72     implementation 'com.google.android.gms:play-services-location:18.0.0'
73     implementation 'androidx.camera:camera-camera2:1.1.0-alpha09'
74
75     // para usar las imágenes en el recyclerview desde firestore
76     implementation 'com.github.bumptech.glide:glide:4.12.0'
77     //Para usar redondeo de imágenes y demás
78     implementation 'jp.wasabeef:glide-transformations:4.1.0'
79 }
```



- Se inicia con el proceso para almacenar la información de los lugares

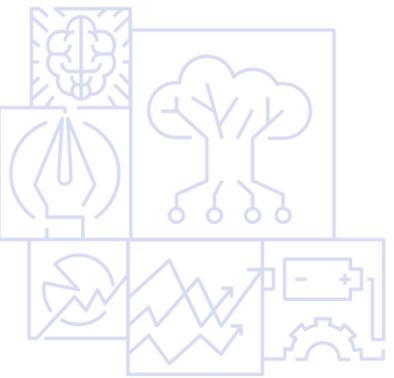
- ¿Qué debemos almacenar del lugar?  
¿Cómo será la clase Lugar.kt



- Se crea la clase Lugar.kt dentro del paquete model



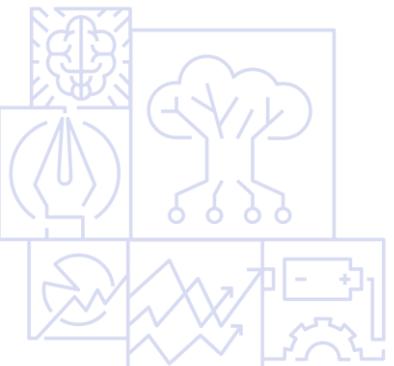
```
1 package com.isc.lugares.model
2
3 import android.os.Parcelable
4 import androidx.room.ColumnInfo
5 import androidx.room.Entity
6 import androidx.room.PrimaryKey
7 import kotlinx.parcelize.Parcelize
8
9 @Parcelize
10 @Entity(tableName = "lugar")
11 data class Lugar(
12     @PrimaryKey(autoGenerate = true)
13     val id: Int,
14     @ColumnInfo(name = "nombre")
15     val nombre: String?,
16     @ColumnInfo(name = "correo")
17     val correo: String?,
18     @ColumnInfo(name = "telefono")
19     val telefono: String?,
20     @ColumnInfo(name = "latitud")
21     val latitud: Double?,
22     @ColumnInfo(name = "longitud")
23     val longitud: Double?,
24     @ColumnInfo(name = "altura")
25     val altura: Int?,
26     @ColumnInfo(name = "rutaAudio")
27     val rutaAudio: String?,
28     @ColumnInfo(name = "rutaImagen")
29     val rutaImagen: String?
30 ) : Parcelable
```



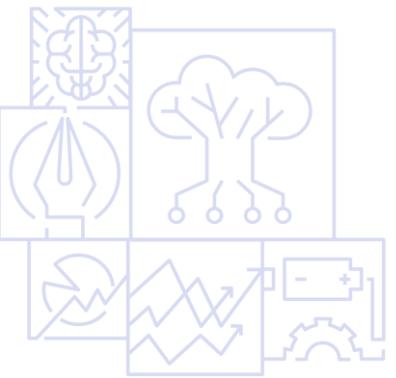
- Se crea la interface LugarDao en el paquete data



```
LugarDao.kt x
1 package com.isc.lugares.data
2
3 import androidx.lifecycle.LiveData
4 import androidx.room.*
5 import com.isc.lugares.model.Lugar
6
7 @Dao
8 interface LugarDao {
9     @Query( value: "SELECT * FROM Lugar")
10    fun getAllData() : LiveData<List<Lugar>>
11
12    @Insert(onConflict = OnConflictStrategy.IGNORE)
13    suspend fun addLugar(lugar: Lugar)
14
15    @Update
16    suspend fun updateLugar(lugar: Lugar)
17
18    @Delete
19    suspend fun deleteLugar(lugar: Lugar)
20 }
```



- Se crea la clase Lugar en el paquete data



```
LugarDatabase.kt
1 package com.isc.lugares.data
2
3 import androidx.room.Database
4 import androidx.room.Room
5 import androidx.room.RoomDatabase
6 import com.isc.lugares.model.Lugar
7
8 @Database(entities=[Lugar::class], version = 1, exportSchema = false)
9 abstract class LugarDatabase: RoomDatabase() {
10     abstract fun lugarDao(): LugarDao
11
12     companion object {
13         @Volatile
14         private var INSTANCE: LugarDatabase? = null
15
16         fun getDatabase(context: android.content.Context): LugarDatabase {
17             val tempInstance = INSTANCE
18             if (tempInstance != null) {
19                 return tempInstance
20             }
21             synchronized(lock: this) {
22                 val instance = Room.databaseBuilder(
23                     context.applicationContext,
24                     LugarDatabase::class.java,
25                     name: "lugar_database"
26                 ).build()
27                 INSTANCE = instance
28                 return instance
29             }
30         }
31     }
}
```

- Se crea la clase **LugarRepository** en el paquete repository



LugarRepository.kt

```
1 package com.isc.lugares.repository
2
3 import androidx.lifecycle.LiveData
4 import com.isc.lugares.data.LugarDao
5 import com.isc.lugares.model.Lugar
6
7 class LugarRepository(private val lugarDao: LugarDao) {
8     val getAllData: LiveData<List<Lugar>> = lugarDao.getAllData()
9
10    suspend fun addLugar(lugar: Lugar) {
11        lugarDao.addLugar(lugar)
12    }
13
14    suspend fun updateLugar(lugar: Lugar) {
15        lugarDao.updateLugar(lugar)
16    }
17
18    suspend fun deleteLugar(lugar: Lugar) {
19        lugarDao.deleteLugar(lugar)
20    }
21}
```

Un repositorio es una clases abstracta para acceder a múltiples fuentes de datos. No es parte de la Librería de Arquitectura de Componentes, se sugiere como una buena práctica para separan la codificación y la arquitectura.

- Se crea la clase LugarViewModel en el paquete viewmodel

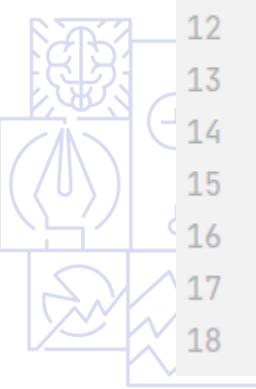


```
LugarViewModel.kt x
11 class LugarViewModel(application: Application): AndroidViewModel(application) {
12     val getAllData: LiveData<List<Lugar>>
13     private val repository: LugarRepository
14     init {
15         val lugarDao = LugarDatabase.getDatabase(application).lugarDao()
16         repository = LugarRepository(lugarDao)
17         getAllData = repository.getAllData
18     }
19     fun addLugar(lugar: Lugar) {
20         viewModelScope.launch(Dispatchers.IO) { repository.addLugar(lugar) }
21     }
22     fun deleteLugar(lugar: Lugar) {
23         viewModelScope.launch(Dispatchers.IO) { repository.deleteLugar(lugar) }
24     }
25     fun updateLugar(lugar: Lugar) {
26         viewModelScope.launch(Dispatchers.IO) { repository.updateLugar(lugar) }
27     }
28 }
```



La función de ViewModel es proveer la información hacia al Interfaz del usuario (UI) y mantenerse a pesar de los cambios de la configuración. El modelo actúa como un centro de comunicación entre el repositorio y la UI

## • Los imports de LugarViewModel

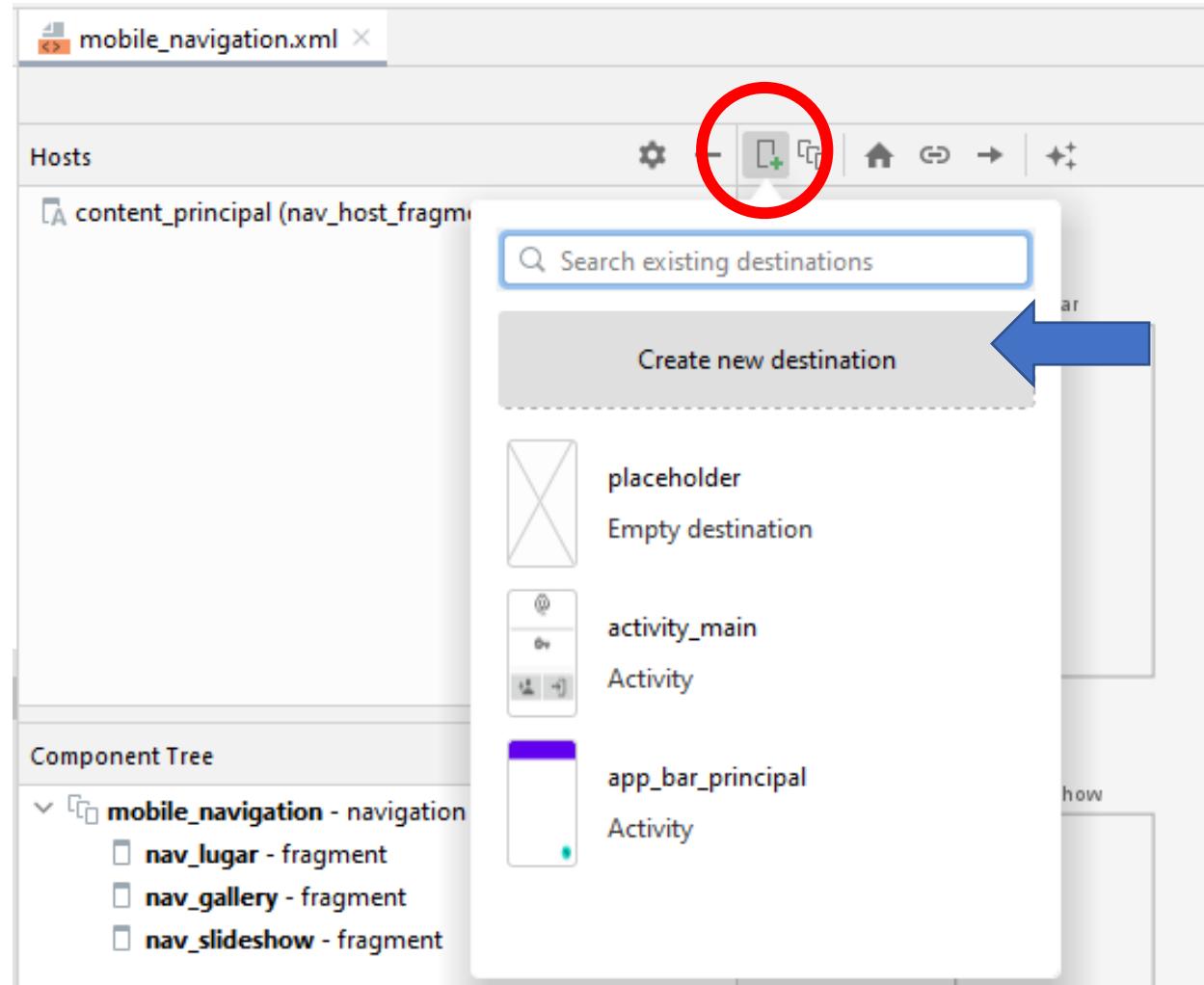


```
LugarViewModel.kt
1 package com.isc.lugares.viewmodel
2 import android.app.Application
3 import androidx.lifecycle.AndroidViewModel
4 import androidx.lifecycle.LiveData
5 import androidx.lifecycle.viewModelScope
6 import com.isc.lugares.data.LugarDatabase
7 import com.isc.lugares.model.Lugar
8 import com.isc.lugares.repository.LugarRepository
9 import kotlinx.coroutines.Dispatchers
10 import kotlinx.coroutines.launch
11 class LugarViewModel(application: Application): AndroidViewModel(application) {
12     val getAllData: LiveData<List<Lugar>>
13     private val repository: LugarRepository
14     init {
15         val lugarDao = LugarDatabase.getDatabase(application).lugarDao()
16         repository = LugarRepository(lugarDao)
17         getAllData = repository.getAllData
18     }
}
```

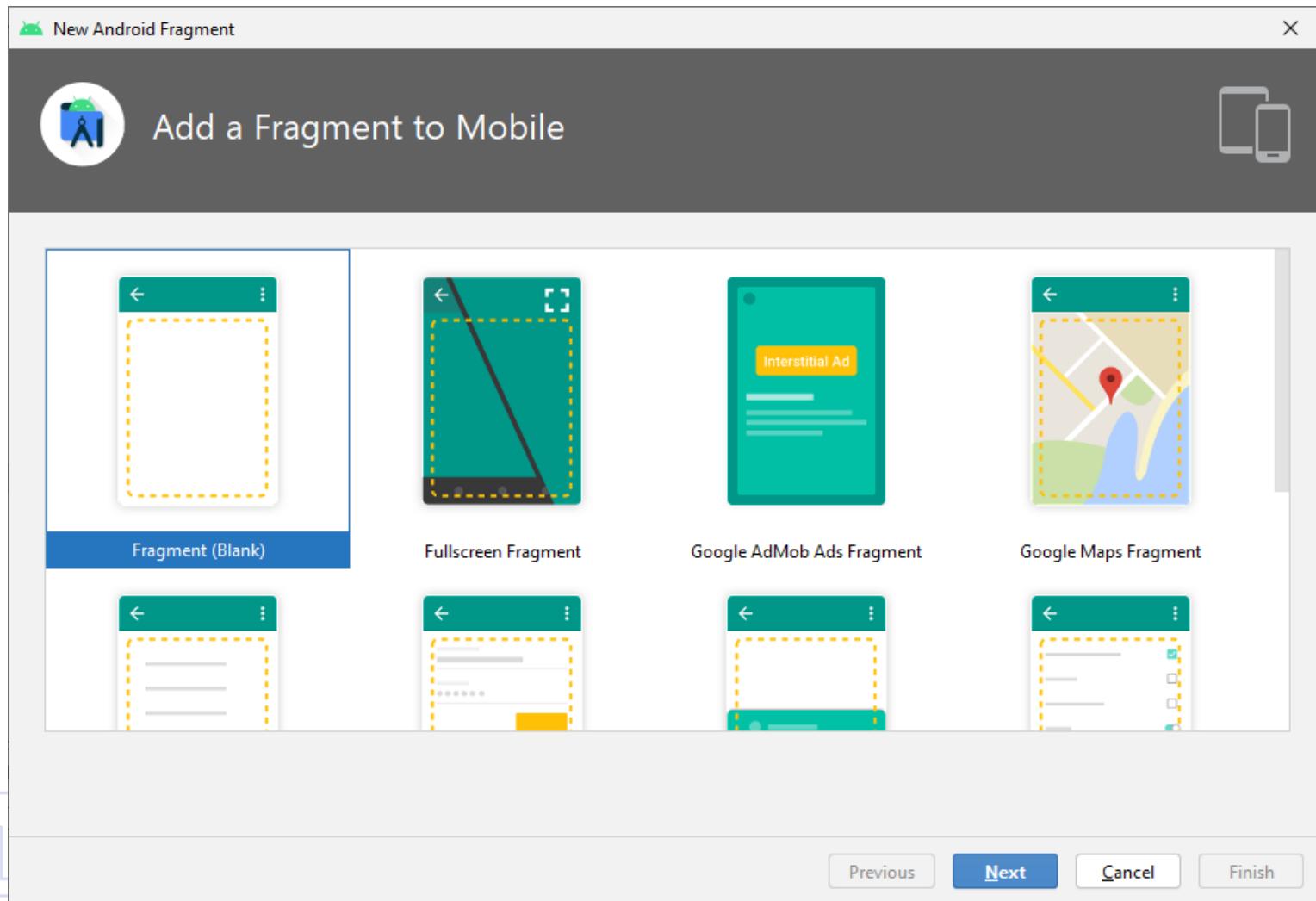
- Se crean los fragmentos y la navegación hacia estos



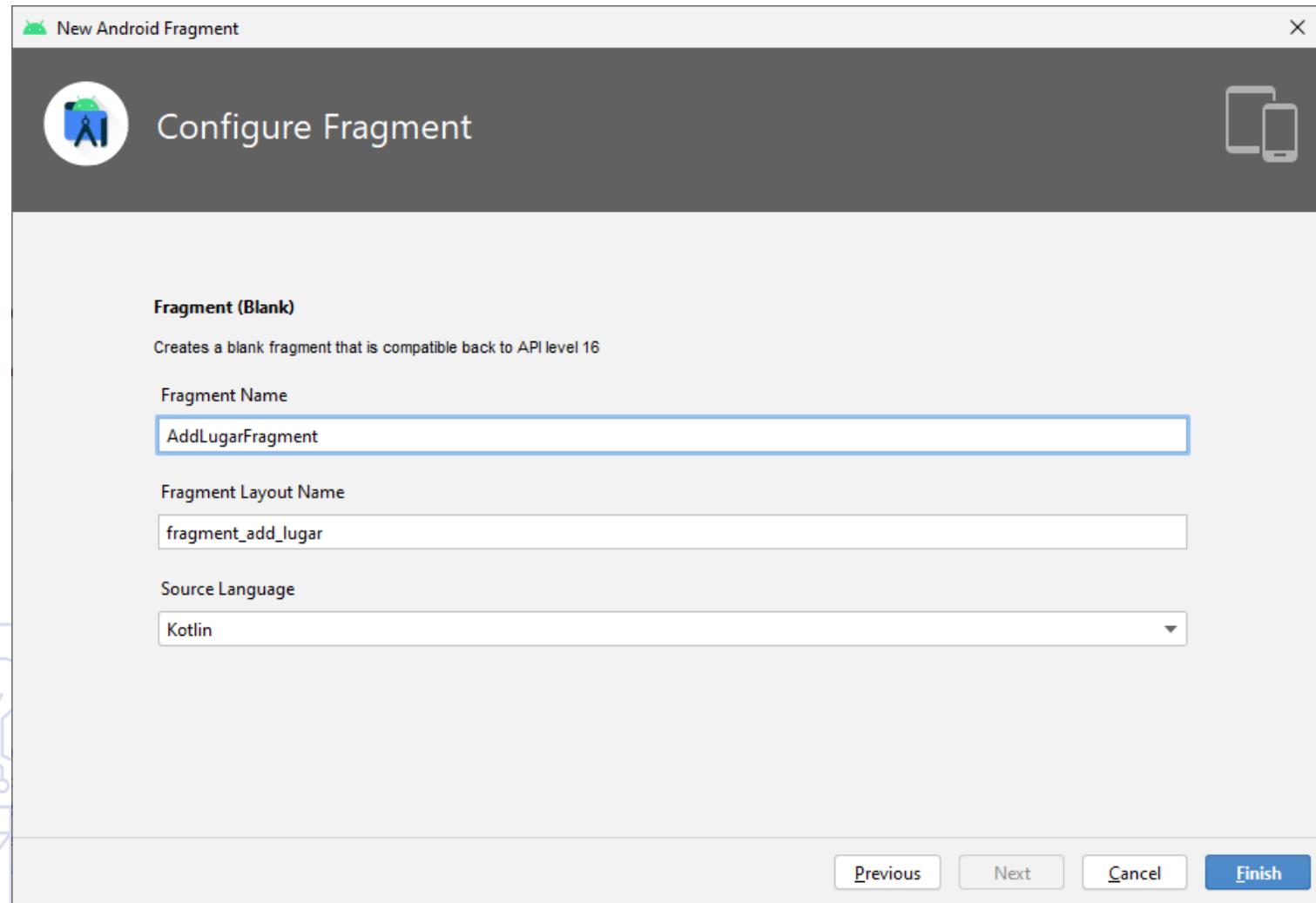
- Se da clic en un nuevo destino



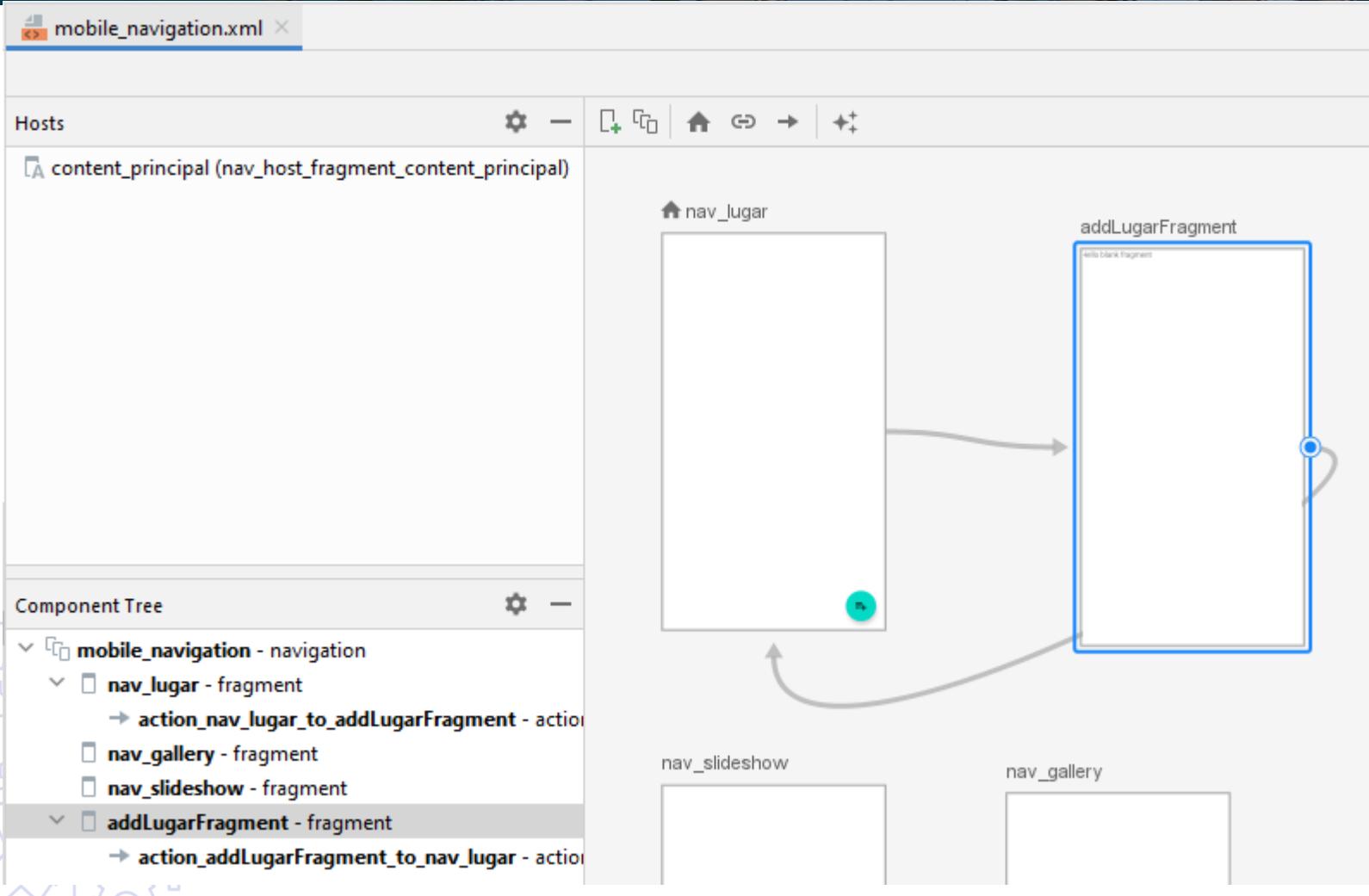
- Se crea un fragmento en blanco



- Se define como AddLugarFragment



- Se definen las navegaciones por medio de los bordes



- Se mueve AddLugarFragment.kt y se modifica mobile\_navigation.xml



The screenshot shows the Android Studio interface. On the left, the project structure is visible, showing the package `com.isc.lugares.ui.lugar` containing the file `AddLugarFragment.kt`, which is highlighted with a red box. The main window displays the XML code for `mobile_navigation.xml`. The code defines three fragments:

```
21     android:label="Gallery"  
22     tools:layout="@layout/fragment_gallery" />  
23  
24 <fragment  
25     android:id="@+id/nav_slideshow"  
26     android:name="com.isc.lugares.ui.slideshow.SlideshowFragment"  
27     android:label="Slideshow"  
28     tools:layout="@layout/fragment_slideshow" />  
29 <fragment  
30     android:id="@+id/addLugarFragment"  
31     android:name="com.isc.lugares.ui.lugar.AddLugarFragment"  
32     android:label="fragment_add_lugar"  
33     tools:layout="@layout/fragment_add_lugar" >
```

The fragment at line 31, which contains the class name `com.isc.lugares.ui.lugar.AddLugarFragment`, is also highlighted with a red box.

- LugarFragment.kt se ajusta así



```
LugarFragment.kt x
  6 import com.isc.lugares.databinding.FragmentLugarBinding
  7
  8 class LugarFragment : Fragment() {
  9     private lateinit var lugarViewModel: LugarViewModel
 10    private var binding: FragmentLugarBinding? = null
 11    private val binding get() = binding!!
 12
 13
 14
 15    override fun onCreateView(
 16        inflater: LayoutInflater, container: ViewGroup?,
 17        savedInstanceState: Bundle?
 18    ): View {
 19        lugarViewModel =
 20            ViewModelProvider(owner: this).get(LugarViewModel::class.java)
 21
 22        binding = FragmentLugarBinding.inflate(inflater, container, attachToParent: false)
 23
 24        return binding.root
 25    }
 26    override fun onDestroyView() {
 27        super.onDestroyView()
 28        binding = null
 29    }
 30}
```

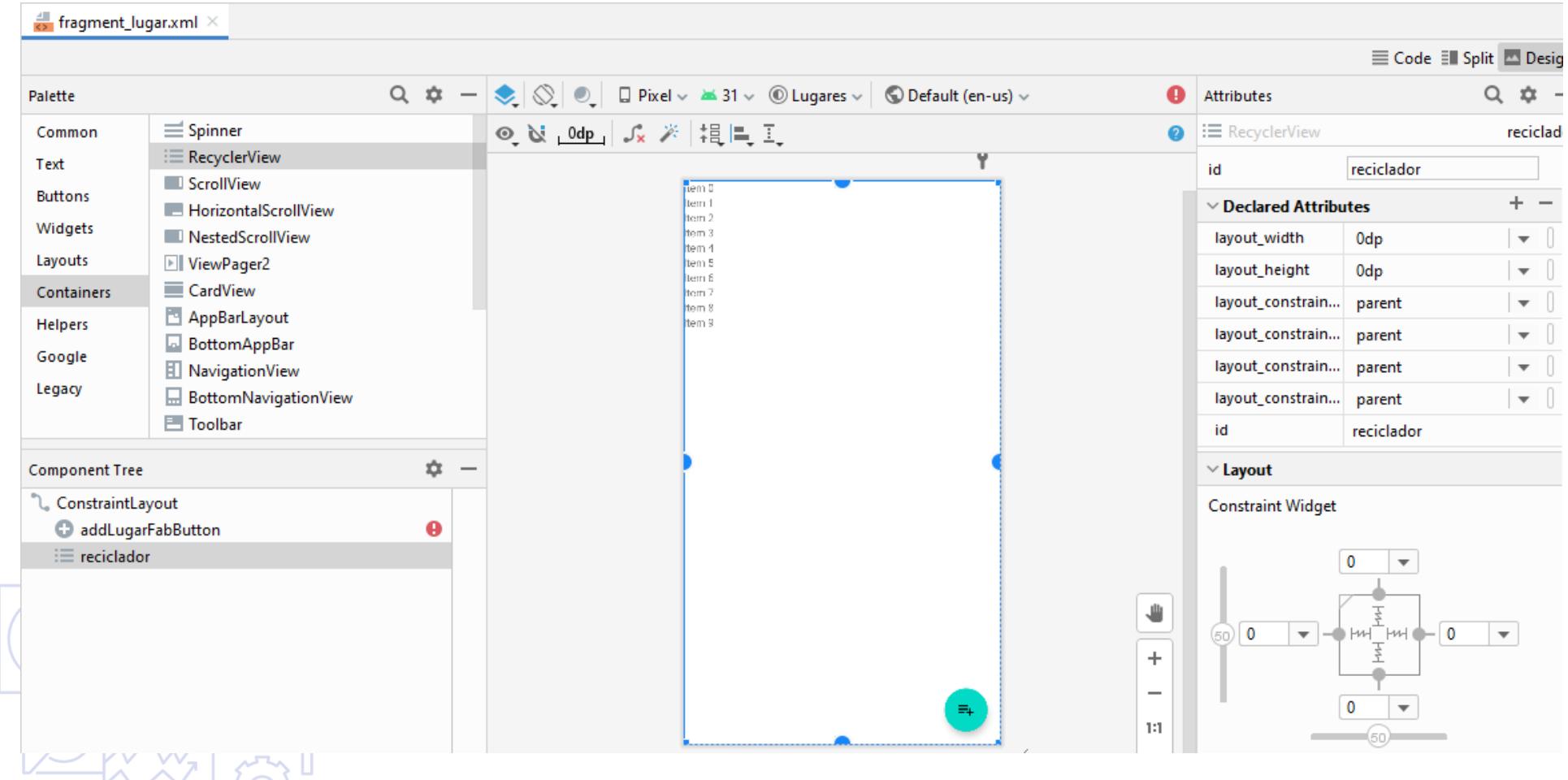


# • Así queda la clase AddLugarFragment

```
AddLugarFragment.kt ×

3 import android.os.Bundle
4 import androidx.fragment.app.Fragment
5 import android.view.LayoutInflater
6 import android.view.View
7 import android.view.ViewGroup
8 import com.isc.lugares.databinding.FragmentAddLugarBinding
9
10 class AddLugarFragment : Fragment() {
11     private var _binding: FragmentAddLugarBinding?=null
12     private val binding get() = _binding!!
13
14     override fun onCreateView(
15         inflater: LayoutInflater, container: ViewGroup?,
16         savedInstanceState: Bundle?
17     ): View {
18         _binding = FragmentAddLugarBinding.inflate(inflater, container, attachToParent: false)
19
20         return binding.root
21     }
22 }
```

# • Así debe quedar fragment\_lugar.xml

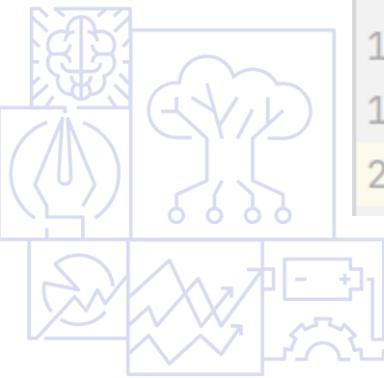


- Se agregan 5 string en el archivo strings.xml

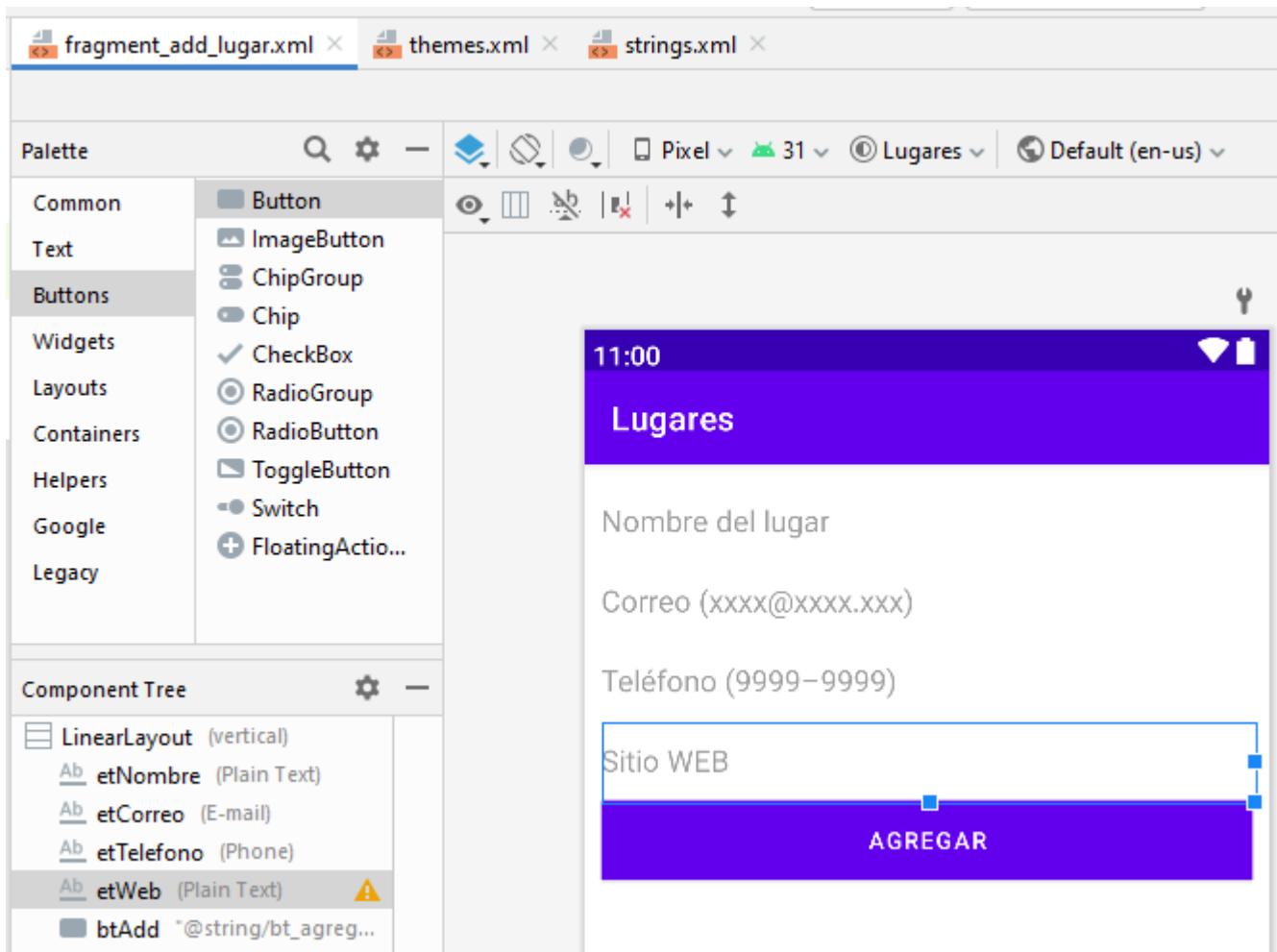


```
strings.xml
Edit translations for all locales in the translations editor.

11   <string name="action_settings">Settings</string>
12   <string name="menu_lugar">Lugares</string>
13   <string name="menu_gallery">Gallery</string>
14   <string name="menu_slideshow">Slideshow</string>
15
16   <string name="msg_nombre">Nombre del lugar</string>
17   <string name="msg_Correo">Correo (xxxx@xxxx.xxx)</string>
18   <string name="msg_Teléfono">Teléfono (9999-9999)</string>
19   <string name="msg_web">Sitio WEB</string>
20   <string name="bt_Agregar">Agregar</string>
```

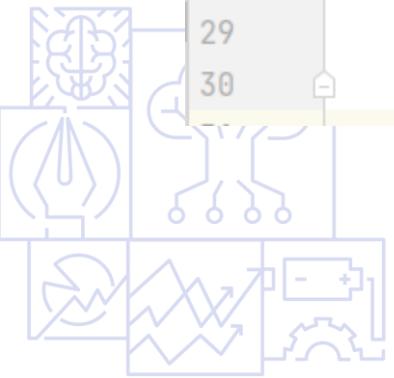


- Así debe quedar fragment\_add\_lugar.xml



## • Actualizamos Lugar\_Fragment.kt

```
LugarFragment.kt x
19         savedInstanceState: Bundle?
20     ): View {
21         lugarViewModel =
22             ViewModelProvider( owner: this).get(LugarViewModel::class.java)
23         binding = FragmentLugarBinding.inflate(inflater, container, attachToParent: false)
24
25         binding.addLugarFabButton.setOnClickListener { it: View!
26             findNavController().navigate(R.id.action_nav_lugar_to_addLugarFragment)
27         }
28
29         return binding.root
30     }
```



- Modificamos AddLugarFragment.kt



```
AddLugarFragment.kt
1 import androidx.lifecycle.ViewModelProvider
2 import com.isc.lugares.databinding.FragmentAddLugarBinding
3 import com.isc.lugares.viewmodel.LugarViewModel
4
5 class AddLugarFragment : Fragment() {
6     private var _binding: FragmentAddLugarBinding?=null
7     private val binding get() = _binding!!
8
9     private lateinit var lugarViewModel: LugarViewModel
10
11
12     override fun onCreateView(
13         inflater: LayoutInflater, container: ViewGroup?,
14         savedInstanceState: Bundle?
15     ): View {
16         _binding = FragmentAddLugarBinding.inflate(inflater, container, attachToParent: false)
17
18         lugarViewModel = ViewModelProvider( owner: this).get(LugarViewModel::class.java)
19
20         binding.btAddLugar.setOnClickListener { insertaLugar() }
21
22         return binding.root
23
24 }
```

- Se actualiza AddLugar.Fragment.kt



```
35     private fun insertaLugar() {
36         val nombre= binding.etNombre.text.toString()
37         val correo= binding.etCorreo.text.toString()
38         val telefono = binding.etTelefono.text.toString()
39         val web=binding.etWeb.text.toString()
40         if (validos(nombre,correo,telefono,web)) {
41             val lugar= Lugar( id: 0, nombre, correo, telefono, web, longitud: 0.0, latitud: 0.0, altura: 0, rutaAudio: "", rutaImagen: "")
42             lugarViewModel.addLugar(lugar)
43             Toast.makeText(requireContext(),getString(R.string.msgLugarAgregado),Toast.LENGTH_LONG).show()
44         } else {
45             Toast.makeText(requireContext(),getString(R.string.msgFaltanDatos),Toast.LENGTH_LONG).show()
46         }
47     }
48
49     private fun validos(nombre: String, correo: String, telefono: String, web: String): Boolean {
50         return !(nombre.isEmpty() || correo.isEmpty() || telefono.isEmpty() || web.isEmpty())
51     }
52 }
53 }
```



• Corremos el proyecto



# Downloads

(Please consider sponsoring us on Patreon 😊)

## Windows

Our latest release (3.12.2) for Windows:

- [DB Browser for SQLite - Standard installer for 32-bit Windows](#)
- [DB Browser for SQLite - .zip \(no installer\) for 32-bit Windows](#)
- [DB Browser for SQLite - Standard installer for 64-bit Windows](#)
- [DB Browser for SQLite - .zip \(no installer\) for 64-bit Windows](#)

**SE  
DESCARGA  
UN VISOR DE  
SQLITE**

<https://sqlitebrowser.org/dl/>

# SE COPIAN LOS TRES ARCHIVOS

Se encuentran dentro de Device File Explorer, dentro de Data -> Data, se busca el paquete -> dentro del paquete está la carpeta databases

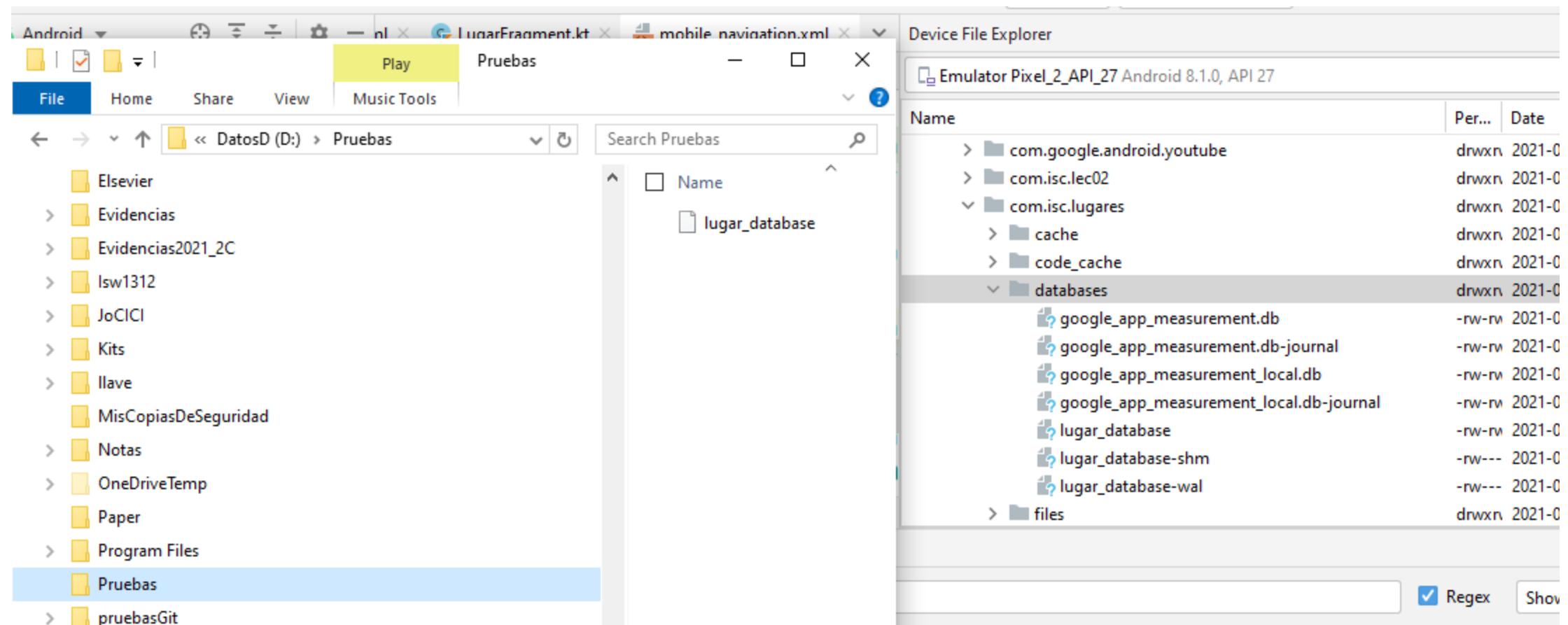
Name	Per...	Date
> com.google.android.youtube	drwxr	2021-09-18 13:15
> com.isc.lec02	drwxr	2021-09-18 13:15
com.isc.lugares	drwxr	2021-09-18 13:15
cache	drwxr	2021-09-18 13:19
code_cache	drwxr	2021-09-18 13:19
databases	drwxr	2021-09-30 22:59
google_app_measurement.db	-rw-rw	2021-09-28 12:41
google_app_measurement.db-journal	-rw-rw	2021-09-28 12:41
google_app_measurement_local.db	-rw-rw	2021-09-30 23:00
google_app_measurement_local.db-journal	-rw-rw	2021-09-30 23:00
lugar_database	-rw-rw	2021-09-30 22:59
lugar_database-shm	-rw---	2021-09-30 22:59
lugar_database-wal	-rw---	2021-09-30 22:59
files	drwxr	2021-09-24 10:30
lib	Irwxr	2021-09-30 22:58
shared_prefs	drwxr	2021-09-30 23:00
com.ustwo.lwp	drwxr	2021-09-18 13:15
jp.co.omronsoft.openwnn	drwxr	2021-09-18 13:15

# SE EJECUTA EL VISOR

This PC > Local Disk (C:) > Program Files > DB Browser for SQLite

<input type="checkbox"/> Name	Date modified	Type	Size
bearer	30/5/2021 14:31	File folder	
extensions	30/5/2021 14:31	File folder	
imageformats	30/5/2021 14:31	File folder	
licenses	30/5/2021 14:31	File folder	
platforms	30/5/2021 14:31	File folder	
printsupport	30/5/2021 14:31	File folder	
styles	30/5/2021 14:31	File folder	
translations	30/5/2021 14:31	File folder	
DB Browser for SQLCipher.exe	2/5/2021 17:30	Application	5 487 KB
<input checked="" type="checkbox"/> DB Browser for SQLite.exe	2/5/2021 17:13	Application	5 448 KB
libcrypto-1_1-x64.dll	25/3/2021 21:34	Application exten...	3 331 KB

Se encuentran los archivos en el dispositivo y se copian en algún lugar



# SE PUEDE APRECIAR LA INFORMACIÓN

DB Browser for SQLite - D:\Pruebas\lugar\_database

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project

Database Structure Browse Data Edit Pragmas Execute SQL

Create Table Create Index Print

Name	Type	Schema
Tables (4)		
android_metadata		CREATE TABLE android_mi
lugar		CREATE TABLE `lugar` (
id	INTEGER	"id" INTEGER NOT NULL
nombre	TEXT	"nombre" TEXT
correo	TEXT	"correo" TEXT
telefono	TEXT	"telefono" TEXT
latitud	REAL	"latitud" REAL
longitud	REAL	"longitud" REAL
altura	INTEGER	"altura" INTEGER
rutaAudio	TEXT	"rutaAudio" TEXT
rutaImagen	TEXT	"rutaImagen" TEXT
room_master_table		CREATE TABLE room_mas
sqlite_sequence		CREATE TABLE sqlite_sequ
Indices (0)		
Views (0)		
Triggers (0)		

DB Browser for SQLite - D:\Pruebas\lugar\_database

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project

Database Structure Browse Data Edit Pragmas Execute SQL

Table: lugar

	id	nombre	correo	telefono	latitud	longitud	altura	rutaAudio	rutaImagen
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	Subway	sub@gmail.com	83898989	0.0	0.0	0		

A blurred background image showing two people sitting at a table in what appears to be a classroom or study area. One person is looking down at a book or paper, while the other is looking towards them. The scene is set against a window with vertical blinds.

**CONTINUAMOS LA PRÓXIMA CLASE**